Enterprise Automation Advancement Model (EAAM)

	Baseline	Beginner	Intermediate	Advanced	Expert
Infrastructure & Environments	•Manual provisioning of servers •Manual configuration of application and security •Basic availability monitoring for application and infrastructure	• Standardized provisioning • Some scripted configuration of application and security •Application logs centralized	•Automated provisioning , may not be tied to a deployment pipeline •Automated configuration for application and security, under source control •Application performance monitoring available	•Provisioning automated as part of pipeline for non-production •Application and security configuration through deployment pipeline •Monitoring tools provide consolidated metrics/dashboards and automatic incident creation	•Provisioning and infrastructure configuration via Infrastructure as Code/Configuration (IaC) as part orchestrated release •Anomaly detection and organization-wide distribution of application metrics
Build & Deploy	•Manual processes for managing source code, deployments and orchestration •Merges are infrequent and difficult	•Code organized into modules or tightly coupled components with early branching for releases •CI server performs builds and triggers unit tests •Deployment scripts run manually	•Application components loosely-coupled with late branching •CI server builds triggered by commit and trigger deployments to test environments •Release pipeline executes most functional and non-functional testing	•Distributed/staged builds tied to pre-tested commits traceable to story, task or defect •Zero-downtime deployments. •Deployments only allowed through pipeline	•Application components can be released independently; every commit becomes a potential release candidate •Automation pipeline is “no touch” until final production sign off (if required)
Data Management	•Test data creation is ad hoc and resource intensive •Manual reporting of process development and operations metrics via ad hoc tools (e.g. excel, etc.)	•Test data sets created via ad hoc dumps from System of Record (SOR), which may be out of date •Application process, development and operations metrics are reviewed manually in enterprise tools	•Use of test data management (TDM) automation to publish data sets •Automated build, test and project metrics available via dashboards and status subscriptions	•The ability to pull/utilize test data sets is fully automated •Operational analytics shared across business unit for all environments	•Testing data loading/reset controllable by consumers of services (self-service) •Dynamic sharing of consolidated application metrics radiated across organization
Test & Quality	•Manual test case creation •Manually documented requirements/acceptance criteria •Manual test execution for all phases •Manualresults logging, defect entry into test management tool, spreadsheets, etc	•Some automated test case creation with library of test cases available for re-use •Limited automated test execution •Defects logged automatically by tools for automated test scripts and manually for non-automated testing	•Acceptance criteria used to generate test cases •Automation of most test phases (i.e. regression, performance, unit, functional, etc.) •Automated defect reporting triggered by failures of functional tests	•Automated test case creation driven from use cases / stories •Event triggered automated testing for all test phases with coverage/ traceability reporting •Some manual UAT. •Defect metrics used to analyze code; baselines and trends drive change	•Model-based testing to establish test cases, corner cases, etc •Event triggered automated testing for all test phases, including all UAT; possible automated business sign-off •Majority of defects caught before code enters test environment; focus shifts to defect prevention rather than detection
Organizational Enablement	•Change management requests and deployment approval processes are manua. •Development communications are ad hoc; no centralized driver	•Non-production deployment approvals automated, some automated change requests •Application development communications process and cadence defined within each group	•Automated creation of change requests for a deployment; some closed by automated tools •Automation engineers in place • Resource rotation enables cross-team learning and behavior changes	•Change tasks closed automatically as part of the deployment pipeline .Developers responsible for their code throughout the deployment pipeline	•Deployment approvals fully automated, with the possible exception of final sign-off for production •DevOps model fully implemented with shared responsibilities for code in production

The Enterprise Automation Advancement Model (EAAM) and survey are intended to be a lightweight, self-diagnostic toolkit to aid application development teams on their journey towards continuous delivery and enable them to:

- measure progress, via a bias-free mechanism, against all facets of automation advancement
- drive 2016 automation planning and activity prioritization based on leading industry practices
- identify additional opportunities to expand and accelerate automation capabilities

The EAAM is unique in that it is:

- Technology Agnostic – applies to all technology platforms (mainframe, midrange/distributed, web, mobile)
- Scalable – scalable to all readiness stages, including areas where teams may be at baseline levels
- Enterprise Relevant – tailored toward enterprise environments, where challenges such as mainframe development and test data management are more prevalent